MATEC Web of Conferences **76**, 02009 (2016)
CSCC 2016

DOI: 10.1051/matecconf/20167602009

Using improved analytical programming algorithm for effort estimation in software engineering

Tomas Urbanek[1,a], Zdenka Prokopova[1], Radek Silhavy[1], and Ales Kuncar[1]

[1] *Tomas Bata University in Zlin, Faculty of Applied Informatics, Namesti T.G.Masaryka 5555, 760 01 Zlin, Czech Republic*

Abstract. This paper evaluates the usage of improved analytical programming algorithm for software effort estimation. The new model was generated by improved analytical programming and it was tested and compared with Karner's model to assess its properties. Least Absolute Deviation and random sub-sampling cross validation were used to assess the reliability to this experiment. The experimental results shows that the new model generated by analytical programming outperforms the Karner's equation about 40 %. Moreover, this work shows that improved analytical programming algorithm is feasible method for calibrating Use Case Points method. All results were evaluated by standard approach: visual inspection and statistical significance testing.

1 Introduction

Effort estimation is the activity of predicting the amount of effort required to complete a software development project [1].The reason for this research is to find effective method, which helps to project managers estimate effort more accurate. Accurate and consistent predictions are crucial point in project management for effective planning, monitoring and controlling the software development cycle. Project managers also used these predictions for better management decisions.

There are a great deal of factors which influencing the final prediction; for instance, the size of development team, used programming language, the complexity of requirements and other factors. One of the most substantial factor is human factor. In this point of view, the use of artificial intelligence could be a promising way for effort estimation in software engineering. Nowadays, using of artificial intelligence is very common in this research field. Hence, in this article the method of improved analytical programming for effort estimation is investigated.

In recent time we published study that examined a selection of fitness function for analytical programming and it was found that the best fitness functions are LAD, MSE and very common MMRE [2]. Therefore, in this work we use a LAD as accuracy measurement. This article is evolution of the published study [3]. In this article we investigated the one specific model from generated population. However, there were no statistical evidence of the better accuracy. This paper provides the comparison of Karner's model with models generated by the improved analytical programming algorithm [4]. In our best knowledge no previous study has investigated this improved analytical programming algorithm on real world dataset. Therefore, this study makes a major contribution to research of Use Case Points method, while the improved analytical programming algorithm is used.

[a]Corresponding author: turbanek@fai.utb.cz

2 Related work

Despite of a great deal of effort of scientists and software engineers, there is still no optimal and effective method for every software project. Some work has been done to enhance the effort estimation based on the Use Case Points method. These enhancements cover the review and calibrating the productivity factor such as the work of Subriadi et al. [5]. Another enhancement could be the construction investigation and simplification of the Use Case Points method presented by Ochodek et al. [6]. The recent work of Silhavy et al. [7] suggest a new approach " automatic complexity estimation based on requirements ", which is partly based on Use Case Points method. Very promising way is a research of Kocaguneli et al. [8], this paper shows, that ensemble of effort estimation methods could provide better results than a single estimator.

2.1 The Use Case Points method

This effort estimation method was presented in 1993 by Gustav Karner[9]. It is based on a similar principle to the function point method. Project managers have to estimate the project parameters to four tables. These tables are as follows:

- Unadjusted Use Case Weight (UUCW)
- Unadjusted Actor Weight (UAW)
- Technical Complexity Factor (TCF)
- Environmental Complexity Factor (ECF)

Due to the aims of this paper, the detailed description of well known Use Case Point method basic principles is insignificant and hence omitted. Please refer to [9], [2] for more detailed description of the Use Case Point method.

2.2 Differential evolution

Differential evolution is an optimization algorithm introduced by Storn and Price in 1995, [10]. This optimization method is an evolutionary algorithm based on population, mutation and recombination. Differential evolution is easy to implement and has only four parameters which need to be set. The parameters are: Generations, NP, F and Cr. The Generations Parameter determines the number of generations; the NP Parameter is the population size; the F Parameter is the weighting factor; and the Cr Parameter is the crossover probability [11]. In this research, the differential evolution is used as an individual generator for analytical programming.

2.3 Analytical programming

Analytical programming (AP), is a symbolic regression method. The core of analytical programming is a set of functions and operands. These mathematical objects are used for the synthesis of a new function. Every function in the analytical programming set core has its own varying number of parameters. The functions are sorted according to these parameters into General Function Sets (GFS). For example, GFS_{1par} contains functions that have only 1 parameter – e.g. $sin()$, $cos()$, or other functions. AP must be used with any evolutionary algorithm that consists of a population of individuals for its run [12], [13]. In this paper, we use improved analytical programming algorithm which is described in [4].

3 Problem statement

In this section the design of the research question is provided and can be outlined as follows:

- RQ-1: Analysing the effectiveness of models synthesised by improved analytical programming algorithm with Karner's model.

The research question (RQ-1) aims to obtain an insight on the estimation accuracy of improved analytical programming algorithm and understand the actual effectiveness of this technique with respect to the estimates by standard Use Case Points method. For this reason, the productivity factor will be set to the standard value of 20. Then the estimates will be calculated by improved analytical programming algorithm and then compared with standard Use Case Points method. One sample t-test and descriptive statistics is utilised to asses the statistical evidence of the effectiveness of this technique.

4 Experiment planning

The proposed experiment can be seen in the Figure 1. In this experiment we used repeated random sub-sampling cross validation. In one loop was generated one equation which was then verified on the rest of the dataset. The process begins with a cycle that loops through the 100 random sub-samples. In the data preparation loop, the random subsampling cross validation was used to split the dataset into two distinct sets (training set 60 % and testing set 40 %). In the second loop, the differential evolution process starts to generate an initial population. Analytical programming then uses this initial population to synthesize a new function with constants resolved. After that, the new function is evaluated by the least absolute deviation measure. If the termination condition, which can be the number of population, is met, one can assume that one has an optimal predictive model, and this model is then evaluated by the calculation of the least absolute deviation on the testing set. Then, the results are saved to file for further analysis.

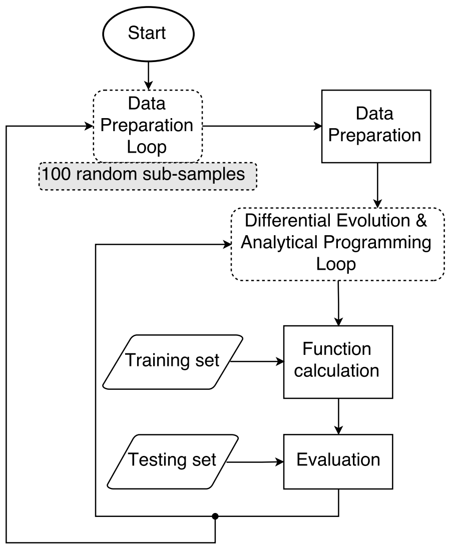

Figure 1. Diagram of proposed experiment.

4.1 Dataset

The data for this study was collected using document review. There are Use Case Points data from 86 software projects and five values for each project: UUCW, UAW, TCF, ECF and actual effort. The distribution of this dataset can be seen on Figure 2. On this figure can be seen bimodal distribution with two distinct peeks. First peek about roughly 700 man/hour and the second peek in roughly 1800 man/hour.

Table 1 shows the analytical programming set-up. The number of leafs (functions built by analytical programming can be seen as trees) was set to 20, which can be recognized as a relatively high value. However, one needs to find the model that will be more accurate than the Karner's model. There is no need to generate short and easily memorable model, but rather, model that will be more accurate.

Table 2 shows the set-up of differential evolution. The best set-up of differential evolution is the subject of further research.

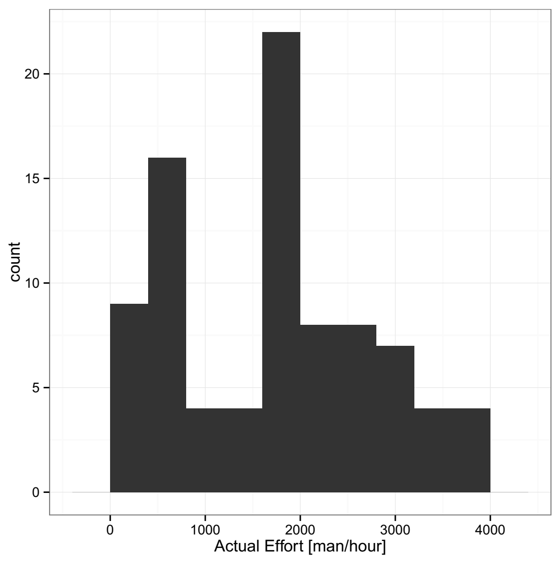

Figure 2. The distribution of actual efforts in dataset.

Table 1. Set-up of analytical programming.

Parameter	Value
Number of leafs	30
GFS - functions	Plus, Subtract, Divide, Multiply, Power, Sqrt, Abs, Sin, Cos
GFS - constants	UUCW, UAW, TCF, ECF, K
Constant K range	0-10

Table 2. Set-up of differential evolution.

Parameter	Value
NP	45
Generations	670
F	0.2
Cr	0.9

4.2 Fitness function

The new model built by the analytical programming method contains the following parameters: UUCW, UAW, TCF and ECF. There is no force applied to the analytical programming that the models have to contain all of these parameters.

$$LAD = \sum_{i=1}^{n} |y_i - \hat{y}_i| \qquad (1)$$

,where n is equal to the number of projects in training set, $\hat{y}_i$ is prediction,y_i is actual effort

The Equation 1 is used for optimization task. When the Least Absolute Deviation result is closer to zero then the accuracy of the proposed model is higher.

5 Results

In this section, we present the result of our study. Descriptive statistical analysis and one sample t-test was utilized to describe research results. All the calculations was performed with repeated random sub-sampling cross validation on 86 software projects.

The LAD value for the Karner's equation on the whole dataset is 75744 man/hour. Hence the average error on one software project is 880 man/hour.

Table 3 provides the calculation of 5 point summary of least absolute deviation on training set, testing set and on the sum of training and testing set. The most interesting value is 41209 man/hour as a maximum of the sum of testing and training dataset with the comparison of Karner's equation LAD error. Also can be noted that medians and means of the calculation are very similar. This is also true for inter quartile ranges.

5.1 Statistical test

One sample t-test was conducted to provide the evidence that the new new created models had statistically lower LAD error than Karner's equation.

```
One Sample t-test
data: data$Complete
t = -184.5056, df = 99, p-value < 2.2e-16
Alternative hypothesis: true mean is less than 75744
95 percent confidence interval:
-Inf, 34681.93
Sample estimates:
mean of x
34309.05
```

As can be seen the p-value of the t-test has been $2.2 * 10^{-16}$. The null hypothesis that the Karner's equation had less true mean is not accepted. Hence, there is a statistical evidence that the new method produce more accurate models.

6 Discussion

This section begun with answering the question of whether improved analytical programming outperformed the standard UCP equation. This question is answered in the result section. If the productivity factor and the whole UCP method is set to default values, there is a possibility, that model built by analytical programming outperform the standard UCP equation.

There is question (RQ-1), which must be answered. For answering this question we need to study table 3 and statistical t-test from result section. From table 3 could be seen that, the value of maximum (statistically worst model) of the sum of testing and training dataset is 41209 man/hour. This is much lower LAD error than for the Karner's equation. The mean and median model have had values roughly 34000 man/hour. For this models is average LAD error for one software project 395 man/hour.

Table 3. The statistical summary of synthesized models.

	LAD Training set [52 projects]	LAD Testing set [36 projects]	LAD Training+Testing set
Min.	15297	9424	26990
1st Qu.	18675	12994	33358
Median	20027	14040	33849
Mean	20008	14301	34309
3rd Qu.	21331	15515	35316
Max.	24236	20366	41209

Hence, there is a 50% probability that, the improved analytical programming algorithm generate a 2.5 times more accurate equation than the standard UCP equation. The interesting result is the maximum values in table 3. There is no equation, which reaches the penalization. This can be seen as a property of the usage of the improved analytical programming algorithm. The t-test in result section also provides evidence of the accuracy improvement of generated equations.

Evidences provided by these statements could be probably false, when the productivity factor will be set to the optimal value. However the optimal value for productivity factor is not known and therefore in this paper we used standard productivity factor. When the productivity factor is set to standard value the new equations outperform the Karner's equation.

7 Conclusion

The current study discover that the prediction of effort estimation by improved analytical programming can be seen as a feasible method. However, this statement is true if and only if the UCP method is not optimized. The equations, which outperforms the Karner's equation in average 44 %, could be produced by this technique. The findings of this study have a number of important implications for future research of the using of the improved analytical programming as an effort estimation technique. More research is required to determine the efficiency of analytical programming for this task.

Acknowledgement

This work was supported by Internal Grant Agency of Tomas Bata University in Zlin under the project No. IGA/FAI/2016/035.

References

[1] J.W. Keung, Software Engineering Conference, 2008. APSEC '08. 15th Asia-Pacific pp. 495–502 (2008)

[2] T. Urbanek, Z. Prokopova, R. Silhavy, V. Vesela, SpringerPlus (2015)

[3] T. Urbanek, Z. Prokopova, R. Silhavy, A. Kuncar (2016), Vol. 465, pp. 261–272, ISBN 978-3-319-33620-6

[4] T. Urbanek, Z. Prokopova, R. Silhavy, A. Kuncar, 30th European Conference on Modeling and Simulation pp. 231–236 (2016)

[5] A.P. Subriadi, P.A. Ningrum, Journal of Theroretical and Applied Information Technology **59**, 735 (2014)

[6] M. Ochodek, J. Nawrocki, K. Kwarciak, Information and Software Technology **53**, 200 (2011)

[7] R. Silhavy, P. Silhavy, Z. Prokopova, PLOS ONE **10** (2015)

[8] E. Kocaguneli, T. Menzies, J.W. Keung, IEEE Transactions on Software Engineering **38**, 1403 (2012)

[9] G. Karner, Objective Systems SF AB pp. 1–9 (1993)

[10] R. Storn, K. Price, Vol. 11 (Technical Report TR-95-012, 1995), ISBN TR-95-012

[11] R. Storn, Proceedings of North American Fuzzy Information Processing pp. 519–523 (1996)

[12] I. Zelinka, D. Davendra, R. Senkerik, R. Jasek, Z. Oplatkova, (InTech, Rijeka, 2011), ISBN 978-953-307-171-8

[13] Z.K. Oplatkova, R. Senkerik, I. Zelinka, M. Pluhacek, Computers & Mathematics with Applications **66**, 177 (2013)